AFRL-OSR-VA-TR-2013-0005

(DCT-FY08) Target Detection Using Multiple Modality Airborne and Ground Based Sensors

Avideh Zakhor

Regents of the University of California

March 2013
Final Report

AIR FORCE RESEARCH LABORATORY
AF OFFICE OF SCIENTIFIC RESEARCH (AFOSR)
ARLINGTON, VIRGINIA 22203
AIR FORCE MATERIEL COMMAND

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
17-08-2012	FINAL	4-1-2008 to 11-30-2011

4. TITLE AND SUBTITLE

(DCT-FY08) Target Detection Using Multiple Modality Airborne and Ground Based Sensors

5a. CONTRACT NUMBER

5b. GRANT NUMBER

FA9550-08-1-0168

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Avideh Zakhor

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Regents of the University of California, The University of California Berkeley
2150 Shattuck Avenue, Room 313
Berkeley, CA 94704-5940

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFOSR
875 N Randolph St
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

AFRL-OSR-VA-TR-2013-0005

12. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution A: Approved for public release

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Automated 3D modeling of building interiors is useful in applications such as virtual reality. We have developed architecture and associated algorithms for fast, automatic, photo-realistic 3D models of building interiors. We have developed an ambulatory human operated backpack system made of a suite of sensors such as laser scanners, cameras, orientation measurement units (OMU)s which are used to both localize the backpack, and build the 3D geometry and texture of building interiors. We have developed a number of localization algorithms based on merging laser, camera and OMU sensor information, and compared their performance using a high end IMU sensor which serves as the ground truth. We have shown our average position error for a ½ kilometer path to be 50 cm or 0.1%. Once the backpack is localized, a 3D point cloud can be generated and 3D meshing or plane fitting algorithms are applied to generate texture mapped 3D models. We have generated 3D models for multiple floors of 8 buildings over the past few years. Using 100 surveyed control points, we have found the accuracy of our system to be on average 15 cm for a 750 meter data acquisition path

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(Include area code)*
U	U	U	U		

Reset

FINAL REPORT FOR FA9550-08-1-0168
U.C. Berkeley
Avideh Zakhor

I. MOTIVATION

There is increased demand for 3D models of objects and sites for virtual environment (VE) applications, such as virtual museums, historical sites documentation, mapping of hazardous sites and underground tunnels, mine automation, and modeling of industrial and power plants for design verification and modification. In addition, rapid, automated 3D models of building exteriors and interiors can be extremely useful in urban military operations.

Applications requiring reality-based, accurate, photo-realistic 3D models vary significantly in their requirements. Industrial design and documentation, training, and operations in hazardous environments usually require higher geometric accuracy than marketing or visualization applications. Given the application requirements, selecting and implementing the most efficient method for data collection and modeling is not obvious. Several methods and a variety of sensors exist. They vary significantly in the ability to capture details, cost, accuracy, speed, and ease of use. Selecting the most suitable and efficient method, along with its configuration is challenging because for some of the paradigms, the effects of the many possible configuration parameters remain unknown.

Over the past five years, we have designed, analyzed and developed the architecture together with associated algorithms, for a human operated, portable, 3D indoor modeling system, capable of generating photo-realistic rendering of internal structure of multi-story buildings. Our motivation to focus on a human operated, portable system rather than an autonomous robot stems from speed, robustness, and scalability considerations, as well as the limitations of robots in complex environments such as stairways inside a building, uneven terrain, and dynamic environments. In addition, a human operator can ensure a much more thorough data acquisition for all objects, surfaces, details, and furniture inside a building, than a robot can ever do. In the context of military applications, one can envision a group of soldiers capturing the necessary data to build interior or exterior 3D models while inspecting a multi-story building, or patrolling around a campus filled with buildings.

While at first glance, a human operated system might seem to be easier to develop than a robotic one, the former faces unique and important challenges. First and foremost, a human operated system in which a person carries a backpack full of sensors and equipments, is severely weight and size constrained. Second, unlike a wheeled robotics system which exhibits only three degrees of freedom, namely x, y, and yaw, a human operator exhibits six degrees of freedom: x, y, z, yaw, pitch and roll. Even though it can be argued that pitch, roll and z are small for typical human gait, they cannot be ignored during the localization and model construction process. In addition, since we are interested in modeling complex environments such as staircases, wheel odometry measurements typically used in robotics systems, cannot be applied to our system. Last, but not least, lack of GPS inside buildings, makes it particularly difficult to localize the acquisition system for indoor modeling applications.

The major challenges for an indoor modeling system can be summarized as follows:

- System architecture: What sensor components should be used? What is the spatial, or geometric configuration of the sensors? How many sensors are sufficient to fully capture geometry and texture?
- Localization algorithms: How should the data from heterogeneous sensors be combined to accurately localize the backpack? By localization, we mean recovery of pose, i.e. x,y,z, yaw,

pitch and roll. Without localization, it is not possible to generate a 3D point cloud made of the laser scan returns.

- Geometry modeling: Once localization is completed and a 3D point cloud is generated, what is the best way to model the environment? Traditionally triangulation has been applied to 3D point clouds to result in meshes. However, planar approximation of walls and floors could potentially result in better looking, more artifact free models, even though they could oversimplify the environment or could fail in complex structures such as staircases.
- Texture mapping: Given that the texture for each piece of geometry in a model is captured by multiple cameras, and by multiple frames in a given camera, how does one go about texture selection? Lack of neighborhood consistency can result in disturbing visual artifacts.
- Visualization and rendering: What is the best way to interact with the resulting models? The size of resulting models is generally too large to be viewed with today's renders without Level of Detail (LoD) simplification. Yet LoD simplification needs to be applied in such a way so as to avoid disturbing visual artifacts at the boundaries. Can image based rendering be used to visualize the scene without explicitly constructing a 3D model?

In the remainder of this reportl, we go over each of the above areas. Section 2 deals with system architecture, Section 3 with localization, Section 4 with geometry modeling, Section 5 with texture mapping, and Section 6 with image based rendering.

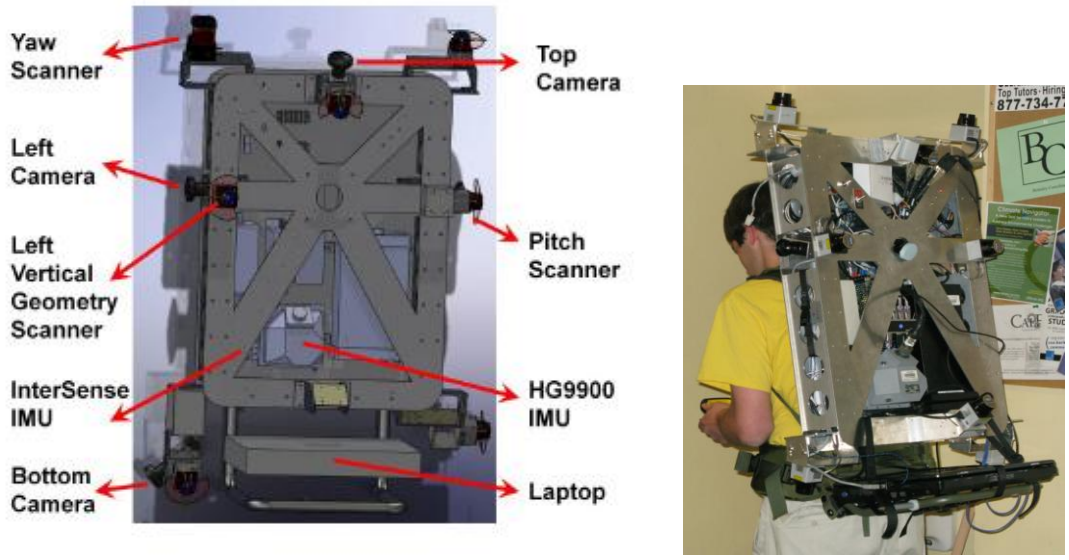
II. System Architecture

Figures 1(a) and 1(b) show the CAD model and an actual picture of our backpack system. We mount three 2D laser range scanners and two Inertial Measurement Units (IMUs) onto our backpack rig, which is carried by a human operator. The yaw scanner is a 40Hz Hokuyo UTM-30LX 2D laser scanner with a 30-meter range and a field of view of 270 degrees. The pitch scanner and left vertical geometry scanner, also known as the roll scanner, are 10Hz Hokuyo URG-04LX 2D laser scanners each with a 4-meter range and a field of view of 240 degrees. These scanners are positioned orthogonal to each other. The InterSense InertiaCube3 IMU is used to provide orientation parameters at the rate of 180 Hz. Another IMU is a strap-down navigation-grade Honeywell HG9900, which combines three ring laser gyros with bias stability of less than 0.003 degrees/hour and three precision accelerometers with bias of less than 0.245mm/ . The HG9900 provides highly accurate measurements of all six degrees of freedom (DOF) at 200Hz and serves as our ground truth. The reasons for using HG9900 only for ground truth are cost, weight, power, and size considerations as well as operational limitations such as frequent zero velocity updates.

We use the laser scanners for both localization and 3D geometry construction. In particular, the left vertical geometry scanner is used to build a 3D point cloud once localization is complete. Similarly, as shown in Section VI, the cameras serve the dual purpose of refining localization as well as texture mapping the resulting models.

III. Localization algorithms

We have developed a series of 3D localization algorithms based on heterogeneous sensors such as laser scanners, IMU and cameras [11]. The outline of this section is as follows. In Section IV-A, we describe localization algorithms based on scan matching. Section IV-B describes ways in which loop closure detection can reduce the overall localization error. Section IV-C outlines our loop closure detection algorithm, and Section IV-D includes performance characterization of our scan matching localization algorithms.



(a) (b)

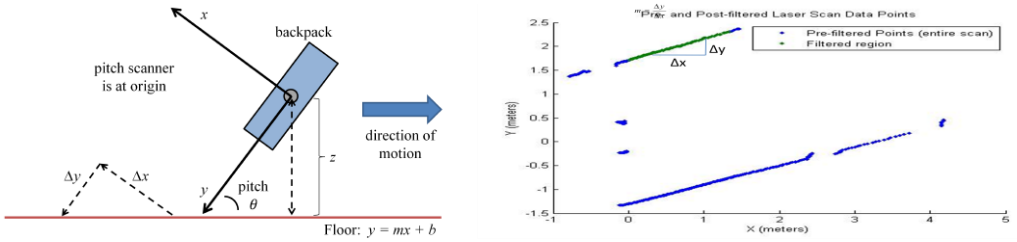
Figure 1. (a) CAD model of the backpack system; (b) the backpack system worn by a human operator.

A. Scan Matching Algorithms

The simplest approach to localization is to use the three orthogonally mounted laser scanners, each assigned to estimate one of the three orientation parameters and two translation parameters via scan matching algorithms such as Iterated Closest Point (ICP). With the backpack worn upright, x represents forward direction, y leftward, and z upward. Referring to Figure 1, the yaw scanner scans the xy plane, the pitch scanner scans the x-z plane, and the left vertical geometry scanner, which is also the roll scanner, scans the y-z plane. Thus, the yaw scanner can resolve yaw rotations about the z axis, the pitch scanner resolves pitch rotations about the y axis, and the roll scanner resolves the roll rotation about the x axis. Assuming that each scanner scans the same plane over time, we can apply scan matching on successive laser scans from each scanner and integrate the translations and rotations obtained from scan matching to recover two translation parameters and one rotation parameter of the backpack over time. Thus, x,y, and yaw are estimated by the yaw scanner, x,z, and pitch by the pitch scanner, and y,z, and roll by the roll scanner. Combining these we can recover all six degrees of freedom; we refer to this as 3xICP algorithm as it runs ICP three times once on each laser scanner [11].

The next simplest alternative is to use the inexpensive, InterSense IMU to estimate pitch and/or roll. The reason for not using yaw estimate from this IMU is that it uses a magnetometer and earth's magnetic field to determine heading. As such, its yaw estimate is usually unreliable due to steel objects distorting the earth magnetic field inside buildings. Thus, we can once again estimate x,y, and yaw by running ICP on the yaw scanner, pitch and roll from the InterSense IMU, and x,z, and pitch by running ICP on the scans from the pitch scanner. This algorithm is referred to as 2xICP+IMU since it runs ICP twice [11].

Finally, it is possible to estimate pitch, roll, and z by assuming the floor is planar, and fitting lines to pitch or roll scans hitting the floor [11]. Pictorially this is shown in Figure 2. In this case, we can run ICP only once on the yaw scanner in order to recover x,y, and yaw, and estimate pitch, roll and absolute z values by fitting lines to the floor portion of the pitch or roll scanner. We refer to this algorithm as 1xICP+planar since it only runs ICP once. A major difference between 1xICP+ Planar algorithm and the other two algorithms described earlier is that the former estimates absolute z values in each step, whereas the latter ones estimate the *change in z* in an incremental fashion from one time step to the next. This is because scan matching between successive pitch or roll scans only provides the change in z and not absolute value of z, thus allowing errors in absolute z to accumulate over time. Implications of this are discussed in more details in Section IV-D.



(a) *(b)*

Figure 2: (a) Side view of the backpack system. The x and y axes show the coordinate system of the pitch laser scanner. The direction of motion assumes that the backpack is worn by a human operator moving forward. (b) A typical scan from the pitch scanner with the green line on top corresponding to a line fit to the floor. Simple geometry can be used to relate pitch to the slope of the green line.

Of course these algorithms can be mixed and matched to generate new ones. For example it is possible to combine 2xICP+IMU and 1xICP+planar in order to arrive at 1xICP+IMU+Planar algorithm. In this case, pitch and roll are estimated by IMU, x, y, and yaw are estimated by the yaw scanner, and z is estimated via the planarity assumption by fitting a line to the floor portion of the pitch scanner.

B. Using Loop Closure to Reduce Errors

Any of the transformation estimation algorithms described above can be used for incremental localization of the backpack from one time instant to the next. By estimating the transformation between poses at consecutive times, we can compose these transformations to determine the entire trajectory of the backpack. However, since each transformation estimate is somewhat erroneous, the error in the computed trajectory can become large over time resulting in loop closure errors. An example of this is shown in Figure 5(a) where the 3xICP algorithm is used in a simple long hallway of about 20 meters to recover the backpack trajectory. As seen, the open loop path can have as much as 8 meters of error in absolute z value. As seen, in Figure 5(a) detecting and enforcing loop closure can eliminate this error.

In Section IV-C we explain how loop closures can be detected. For now, assuming that we have detected time instants in which the backpack is approximately spatially close to where it had visited before, i.e. assuming known loop closures, it is possible to reduce the overall localization error by enforcing loop closures. To do so, we build a graphical model with nodes denoting poses and the edges between the nodes corresponding to transformations between poses obtained via the above set of algorithms. We can also associate a covariance error matrix to each edge, representing the degree of uncertainty of that estimate. In addition, we can estimate the transformation between the beginning node of a loop and the

ending node, i.e. the loop closure node, via any of the above algorithms as long as it is known that the two nodes are close to each other in space. Having constructed this closed loop graph, it is possible to solve an optimization algorithm over the poses associated with each node in order to minimize the overall error. Intuitively, the optimization algorithm takes into account the covariance matrix between every two nodes in order to optimally adjust the pose values at each node.

We choose to use optimization framework TORO to solve this problem [10]. In particular, we supply TORO with a directed graph G = (V; E) with nodes V and edges E where each node in V represents a 6 degrees of freedom (DoF) pose, and each directed edge in E represents a 6-DOF transformation that takes pose at one node to the pose at another node. Each transformation needs to have a covariance matrix specifying its uncertainty [11]. TORO refines pose estimates by using gradient descent to minimize a metric error, thus redistributing the error among the nodes in an optimal fashion. By enforcing loop closure, we are essentially supplying a transformation in graph G that causes G to have a cycle.

C. Loop Closure Detection

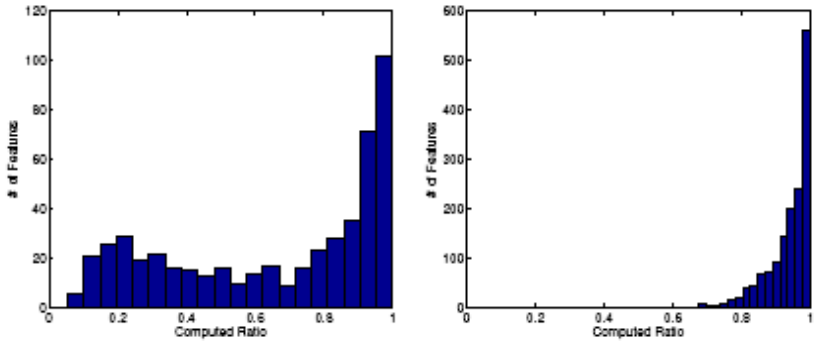
In this section, we describe an algorithm to automatically detect loop closures using camera imagery on the backpack. Our algorithm is a modified version of Newman's FABMAP [12], a probabilistic approach of recognizing places via appearance. Our modifications and extensions to the FAB-MAP algorithm are two-fold. First, after building a vocabulary from training data and converting all scenes into words, we remove words that appear in all, one, or no scenes. This is because we calculate the co-occurrence of words for the Chow Liu tree [14] and words that appear in every image, only one image, or none of the images provide little information in distinguishing images. Second, location prior is left uniform, since performance is largely unaffected [12].

Computing the probability distribution of all images over all locations is considered one trial. We call a match between an image and a location an image pair, since a location originates from an image. This match occurs because the probability of it being a genuine loop closure is higher than a pre-specified threshold; however, in practice it could be either a genuine loop closure or a false positive. In order to emphasize genuine loop closures and recognize false positives, we run 100 trials and record all image pairs with the number of times they appear in the trials. In general, the image pairs with the highest count from FAB-MAP do not necessarily correspond to genuine loop closures. As such, some post-processing is needed to detect genuine loop closures among the top ranked image pair candidates generated by FAB-MAP. Our approach to post processing is to prune image pairs that have appeared the most by using key point matching [13] to determine whether they are correct matches. As shown in [13], correct and incorrect matches have different distributions of the ratio

$$\frac{d(\text{feature, nearest neighbor})}{d(\text{feature, } 2^{\text{nd}} \text{ nearest neighbor})} \qquad (1)$$

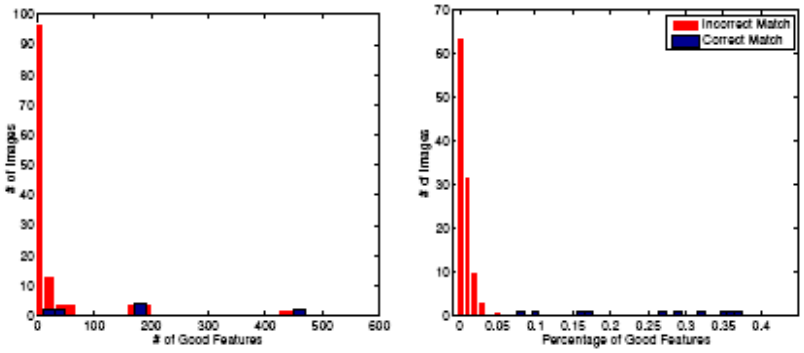
where d(a;b) computes the Euclidean distance between a and b. Figure 3 shows an example of the PDF of the above ratio for all the features in two image pairs resulting from FAB-MAP corresponding to a correct and an incorrect match. As seen, the PDF for genuine and incorrect matches are quite different. Unlike an incorrect match, for a genuine match a significant number and percentage of the features result in the ratio in Equation 1 being smaller than 0.6. Figure 4(a) shows the number of candidate image pairs across all 9 datasets as a function of the number of features satisfying the ratio in Equation 1; each dataset corresponds to a 5 to 10 minute data acquisition and consists of 100 images; there are a total of 13 genuine loop closure pairs for the 9 datasets. As seen, using the absolute number of features is not a reliable indicator of the correctness of a given image pair. By contrast, Figure 4(b) shows the same quantity as a function of the percentage of features satisfying the ratio in Equation 1. As seen, the

percentage of features satisfying Equation 1 can be successfully used to distinguish between correct and incorrect candidate image pairs resulting from FAB-MAP for all 13 loop closures corresponding to 9 datasets.



(a) (b)

Figure 3: The PDF of the ratio of distance of the nearest neighbor to the distance of the second closest neighbor for two image pairs; (a) a correct pair; (b) an incorrect pair resulting from FAB-MAP.



(a) (b)

Figure 4: PDF of (a) the number and (b) the percentage of features with ratio below 0.6.

D. Performance Characterization of Scan-Matching Localization with Loop Closure

We use the 2xICP+IMU and 1xICP+IMU+Planar algorithms described in Section IV-A to characterize the performance of our laser/IMU based localization [11]. In doing so, the loop closures are detected via the approach in Section IV-C, and TORO optimization is applied to the directed graph with transformations resulting from these two algorithms as described in Section IV-B. Both localization methods use scan matching on the yaw scanner to estimate the backpack pose parameters *x*, *y*, and yaw over time. Also, both methods use the InterSense IMU to estimate the backpack roll and pitch over time. For 2xICP+IMU, scan matching on the pitch scanner is used to estimate change in *z* over time. The 1xICP+IMU+Planar method works only in environments with planar floors and fits a line to the floor to estimate absolute *z* at each instant of time.

We test these two algorithms on four datasets: Dataset 1 is a T-shaped corridor intersection that includes a roughly 20 meter segment of a hallway. Datasets 2 and 3 are of a staircase roughly 4.5 meters in height.

Dataset 4 consists of two roughly 15-meter hallway segments connected by a staircase roughly 4.5 meters in height. We first compare 1xICP+IMU+Planar and 2xICP+IMU results on dataset 1 as it is the only dataset with a strictly planar floor and no staircases. For the other datasets, which include staircases, we use 2xICP+IMU localization as it does not require a planarity assumption. Incremental pose errors are compared in the local coordinate frame. Global position and orientation errors are computed in a frame where *x* is east, *y* is north, and *z* is upward. Note that global errors result from accumulated local errors. As such, their magnitude is for the most part decoupled from the magnitude of local errors. In particular, local errors can either cancel each other out to result in lower global errors, or they can interact with each other in such a way so as to magnify global errors.

Global and incremental pose errors using 1xICP+IMU+Planar and 2xICP+IMU for dataset 1 are shown in Figure 5(b). The results are for loop closure detection, followed by TORO optimization. We see that the two methods are comparable with 1xICP+IMU+Planar resulting in significantly lower global *z* error compared to 2xICP+IMU. This is to be expected since in 1xICP+IMU+Planar we estimate the absolute value of *z*, rather than the incremental change in z, at every time step. Thus, the error in *z* does not have a chance to accumulate over time. Nevertheless, since 2xICP+IMU does not make use of a planar-floor assumption, it extends to multi-floor datasets 2, 3, and 4.

Global and incremental pose errors using 2xICP+IMU across all four datasets are shown in Figure 6. Again, the results are for loop closure detection followed by TORO optimization. We see that compared to dataset 1, datasets 2, 3, and 4 corresponding to the multi-floor datasets, have higher error in global roll and yaw. Other errors remain comparable to the single-floor case of dataset 1. Across all datasets, incremental yaw resulting from horizontal scan matching has lower error than incremental pitch and roll resulting from the IMU. The error in pitch and roll by the IMU may be mitigated by calibrating the IMU and removing its bias and drift. The resulting estimated trajectories for all four datasets closely resemble ground truth trajectories, as shown in Figure 7. We used one loop closure for dataset 1 in Figure 7(a) and two loop closures for the remaining three datasets, as input to TORO.

For each dataset, the average position error of the estimated path is reported along with the ground truth path's length in Table I. We find that the average position error relative to the path length for each dataset is small, i.e. around 1% or lower.

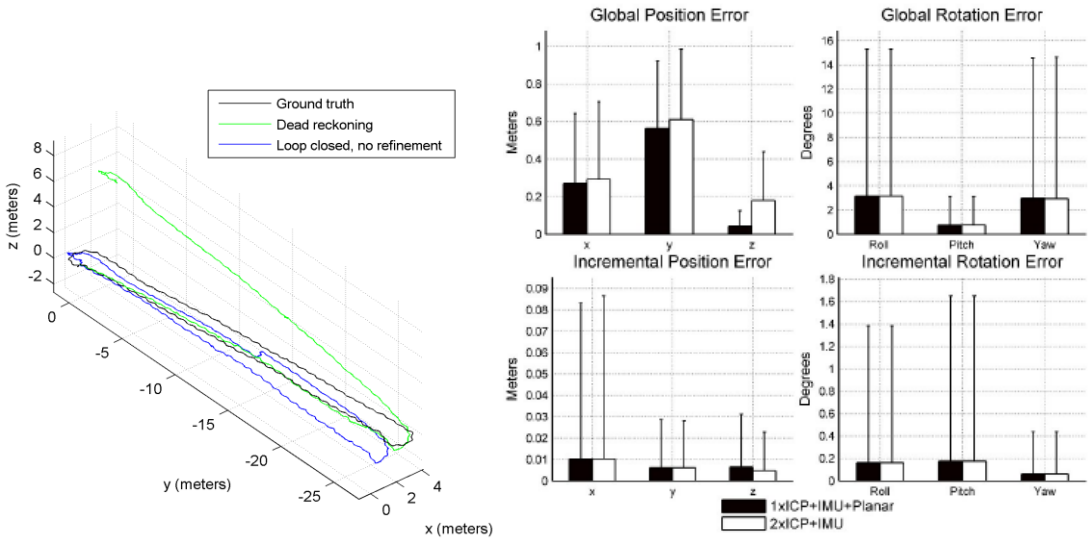


Figure 5. (a) Open loop and closed loop reconstructed paths using 3xICP paths for a simple hallway; (b) Global and incremental RMS error characteristics using 1×ICP+IMU+Planar and 2×ICP+IMU on T shaped corridor of dataset 1. Markers above each bar denote peak errors.

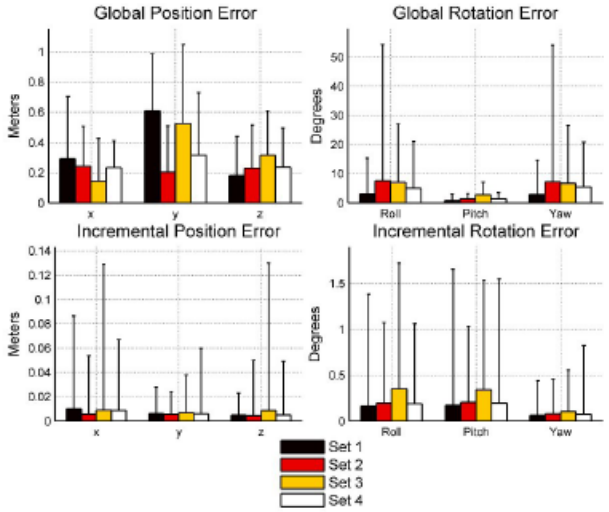


Fig. 6. Global and incremental RMS error characteristics using 2xICP+IMU across all four datasets. Markers above each bar denote peak errors.

Dataset	Path length	Average position error
1	68.73 m	0.66 m
2	46.63 m	0.35 m
3	46.28 m	0.58 m
4	142.03 m	0.43 m

TABLE I
GROUND TRUTH PATH LENGTH VS. AVERAGE POSITION ERROR OF THE ESTIMATED PATH FOR EACH DATASET.

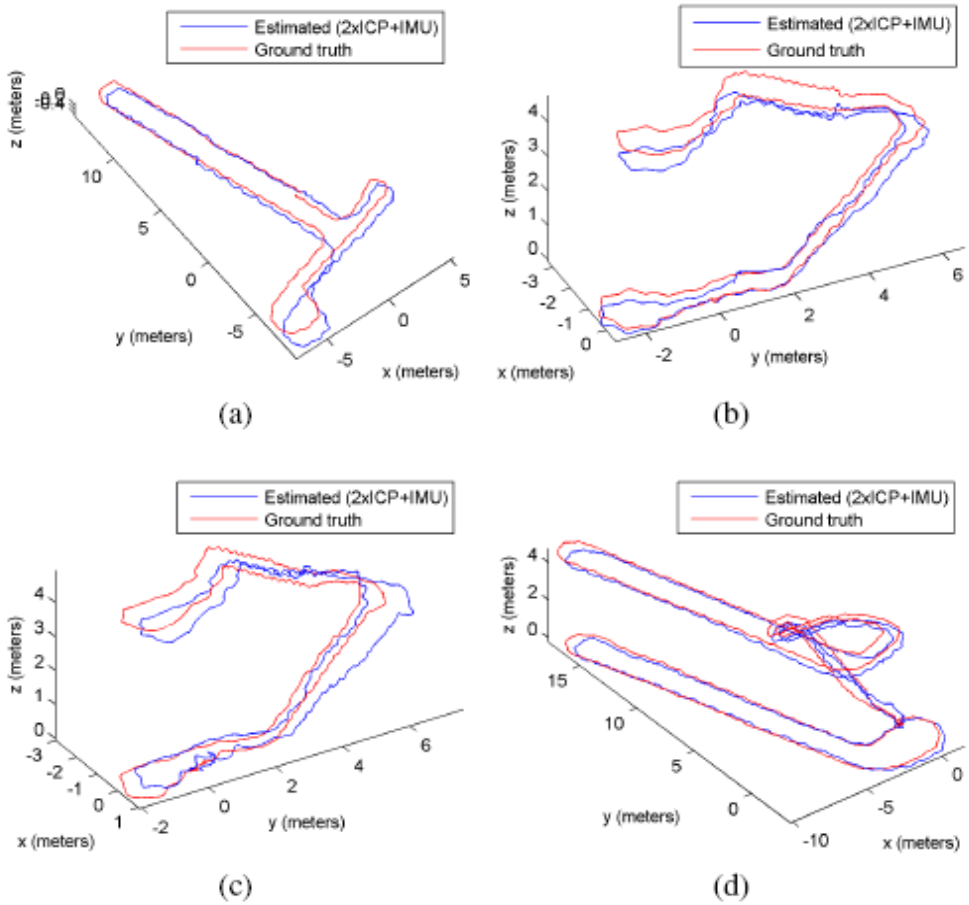
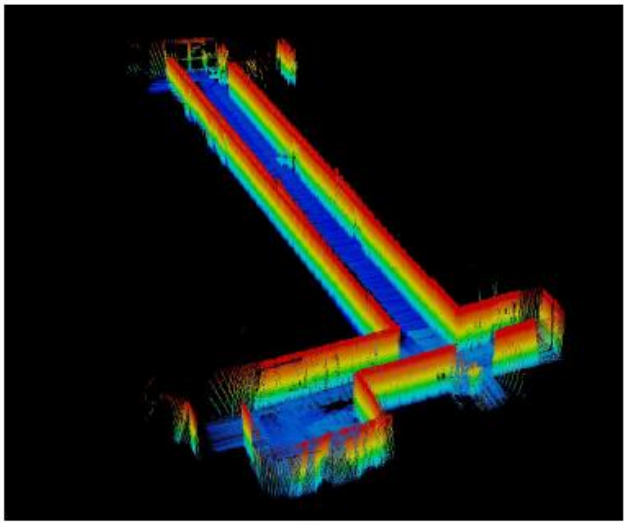


Fig. 7. Estimated trajectories using 2 ICP+IMU plotted against ground truth for: (a) dataset 1; (b) dataset 2; (c) dataset 3; (d) dataset 4. The estimated trajectory using 1xICP+IMU+Planar for dataset 1 is similar to that of 2xICP+IMU.

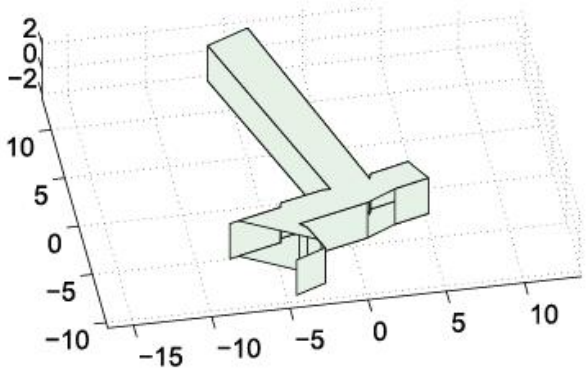
IV. GEOMETRY MODELING

The localization results from Section IV can be applied to the scans from the left vertical geometry scanner in Figure 1 in order to generate a 3D point cloud as shown in Figure 8(a). There are multiple ways to generate a 3D model from this point cloud. One way would be to fit planes to major surfaces of the point cloud as shown in Figures 8(b) and 8(c). The advantage of such an approach is simplicity of the resulting models; the disadvantage is that the resulting models do not always faithfully reproduce the environment under consideration. For example, objects protruding from walls could get leveled to the walls in the process of plane fitting. Also, in complex environments such as staircases, generating models using plane fitting could be quite challenging.

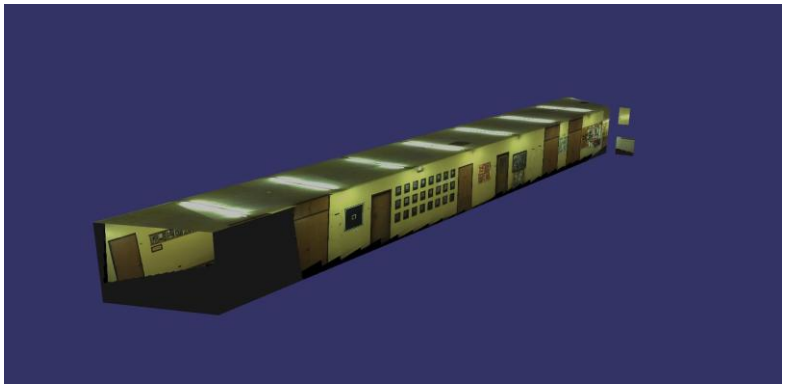
An alternative way to generate a 3D model from a point cloud is to apply the fast triangulation algorithm we developed for outdoor modeling applications[15]. In this approach we take advantage of the order in which the vertical scans are acquired, and the order of the points within each scan so as to generate triangles that are well proportioned, i.e. are not long and skinny. Figure 9 shows such a triangulated and



(a)



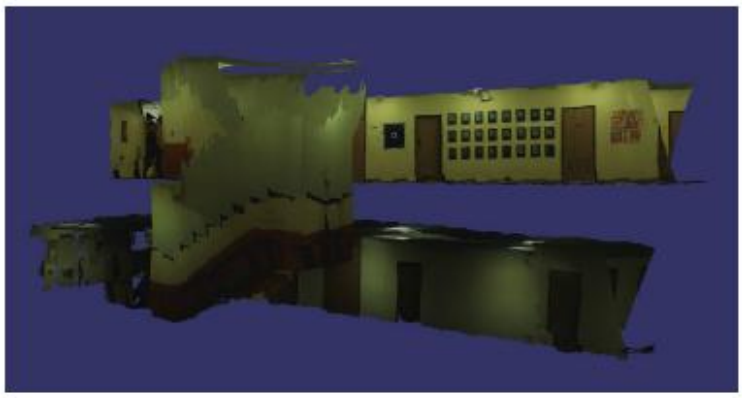
(b)



(c)

Fig. 8. (a) 3D point cloud from laser scanners colored by height and (b) a plane-fitted model of a T-shaped corridor intersection; (c) texture mapped plane fitted model.

texture mapped 3D model for two hallways connected by a staircase corresponding to the trajectory in Figure 7(d). Even though triangulated models represent the details of an environment more faithfully than planar models, generally speaking, they are more prone to visual artifacts.



Fig*ure 9. The 3D model of two hallways connected by a stairwell corresponding to the trajectory shown in Figure 7(d).*

V. TEXTURE MAPPING

Using the pose information provided by the localization algorithms discussed in Section IV-B, we can transform all captured laser scans into a single 3D coordinate frame. Since camera images are acquired at nearly the same time as a subset of the laser scans, nearest-neighbor interpolation of the pose parameters allows us to estimate the pose of every camera image. Therefore, to generate a 3D model, we (i) transform all laser scans from the floor scanner to a single world coordinate frame and use known methods to create a triangulated surface model from ordered laser data [15], and (ii) texture map the model by projecting laser scans onto temporally close images. However, we have found that the laser based localization algorithms alone are not accurate enough for building textured 3D surface models. For example, Figure 10 shows a screenshot of a model created by using the 1xICP+IMU+planar localization results and the resulting texture misalignment. In this section we describe an image based approach to refine the laser/IMU localization results.

Our proposed image based localization refinement is a two step process. First, we estimate the transformations that relate pair-wise camera poses. Second, we obtain a directed graph by adding the estimated pair-wise camera transformations as edges to the graph obtained from the scan matching algorithms. An example of such a directed graph with both sets of transformations is shown in Figure 11. Each small node corresponds to a laser scan, and each large node corresponds to a laser scan and an image acquired at approximately the same time. Small arcs represent transformations resulting from the scan matching algorithms of Section IV-A. Long arcs represent transformations estimated with the image based refinement algorithm to be described shortly. Once a new directed graph with both scan matching and image based edges has been constructed, we perform a round of TORO-based global optimization to redistribute the error among the nodes in order to obtain a set of refined pose estimates [10]. To accomplish this, TORO requires as input the covariance error associated with each transformation shown in Figure 11. As shown later, these new pose estimates lead to better image alignment on the final 3D model because they incorporate both laser based and image based alignment.

We estimate pair-wise image transformation by minimizing Sampson re-projection error [40]. This minimization can only estimate translation up to an unknown scale factor. However, it does provide us with a set of inlier SIFT features; therefore we can estimate the translational scale by aligning triangulated, inlier SIFT features from the side-looking left camera with the 3D laser points acquired by the left vertical geometry scanner. In particular, we perform a multi-resolution search over the unknown scale factor. For each scale, we find the distance between each triangulated SIFT feature and its nearest neighbor in the set of 3D laser points acquired by the floor scanner. We choose the scale factor that minimizes the median distance over the set of inlier SIFT features. The median distance criteria is chosen because it is robust to SIFT outliers that may have been incorrectly identified as inliers.

To obtain the final pose estimates used for 3D modeling, we run a final round of global optimization using TORO on the graph shown in Figure 11 [10]. The pair-wise camera pose transformations are incorporated as edges in the TORO graph. The edges added to the graph span multiple nodes, as shown by the long arcs in Figure 11.

TORO requires that each edge in the graph have a corresponding 6-DOF transformation and a covariance matrix. It is not straightforward to derive a closed-form covariance matrix for our image based transformations since each one is obtained by first minimizing the Sampson re-projection error followed by minimizing the 3D distance between triangulated SIFT features and nearby laser points. However, the covariances for the laser based scan matching algorithms can be computed using Censi's method [17]. Using our ground truth, we can also perform a one time calibration process whereby for each diagonal element of the covariance matrix we compute the relative scale factor between the laser based and the image based localization technique. In particular, over a test run of data acquisition, we first estimate laser based transformations and their covariance matrices without loop closures and then estimate image based transformations without loop closures. For both techniques, we also compute the RMS error for each of the 6 pose parameters by comparison with ground truth provided by the Honeywell HG9900 IMU. For practical situations in which the ground truth is not available, we use this relative scaling factor together with Censi's covariance estimate of the laser based localization to arrive at the covariance estimate for image based localization. In particular, to obtain the covariance matrices for the image based transformations, we average the diagonal covariance matrices from all laser based transformations for a given path and scale them based on the square of the relative RMS error between the two techniques. Intuitively, in the global optimization, this covariance estimation method emphasizes parameters estimated accurately by image based techniques, e.g. translation along the direction of motion, while deemphasizing parameters estimated more accurately by laser based techniques, e.g. yaw.

We have tested the above image based refinement algorithms on a number of data sets. In all tested cases, it does result in a significant improvement in the visual quality of the resulting 3D textured models. Screenshot of the textured 3D model for the scene in Figure 10 is shown in Figure 12. As seen, the misalignments exhibited in Figure 10 are significantly reduced.

The texture mapped 3D models for a number of datasets using the localization results of Section IV together with the above image based refinement algorithm can be downloaded from [18].

Fig. 10. *Screenshot of textured 3D model generated from localization data without image based refinement. Misalignments between textures from different images are apparent.*

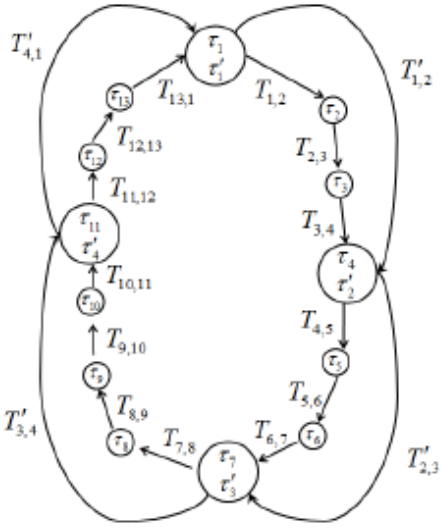


Fig. 11. *Example directed graph for the image based refinement algorithm.*

Fig. 12. *Screenshots of textured 3D model generated with 1×ICP+IMU+planar followed by image based pose estimation and texture blending. The 3D model exhibits good texture alignment in comparison with Figure 10, generated with 1×ICP+IMU+planar.*

VI. VISUALIZATION AND RENDERING

There are two basic ways to visualize and interact with the resulting products from our indoor modeling system; one way is to explicitly build 3D geometric models which the users can interact with within a 3D browser. So far, most of the discussion in this proposal has been centered on this approach. In particular Sections V and VI deal with the explicit 3D geometric construction of models and their texture mapping.

Another way to interact with the resulting data products from our system is image based rendering in which images are rendered at a fast rate from a database based upon viewer's position and orientation. Image based renderers vary based on the relationship between the number of input images and the amount of known geometry [19]–[28]. During the past year, we have developed an image based rendering system which utilizes all data products from the backpack acquisition system: high resolution images from cameras, estimated camera poses from localization algorithms, and the relevant geometry computed from 3D point clouds resulting from laser scanners.

For the T-shaped corridor, each of the 3 cameras in Figure 1 provide 903 images during a five minute walk, and the localization algorithms of Section IV estimate the 6 dimensional pose of a camera for each image. We use the localization results by converting each pose parameter *x*, *y*, *z*, roll, pitch, and yaw into a 3-vector pose representation for each camera on the backpack as shown in Figure 13. Specifically, the estimated camera pose is represented by 3 vectors in 3D: a vector for the position of the camera, a vector for the center of projection of the camera, which represents the orientation, and a normal vector denoted by to represent the up direction. In what follows, we describe various steps of our image based rendering system [40].

A. Selecting the Image to Render

We use a 3-step process to determine which images to render based on the viewer's pose and the set of estimated camera poses. In step 1, if the position vector for a specific camera pose is within a threshold

radius of the viewer's position vector, the associated image is added to set *A* of images to be potentially rendered. The second step involves pruning images in set *A* that are oriented in the wrong direction to obtain set *A'*. The dot product between the viewer's orientation vector and the camera's orientation vector provides a metric for how close a given image is to the viewer's image plane. The resulting set of images *A*' is close to the viewer in both orientation and position; the "best" image in set *A'* is chosen to be the one with the closest position vector to the viewer's. With a tight dot product threshold in step 2, the chosen image has a camera position closest to the viewer's and an image plane in the same direction as the viewer's orientation with minimal deviation. The sets of images *A* and *A'* and the "best" image are shown in Figure 14.

B. Image Mosaicing for Increased Field of View

Similar to most image based renderers, our image based renderer uses matching features between images to mosaic images together to provide an increased field of view [29]. In doing so, we inherently assume that close by images are related to each other by a 3x3 homography. We determine proximity of images by taking advantage of camera pose information as derived from backpack localization results described in Section IV.

Due to the large unstructured data set input to the the renderer, we have taken various steps to optimize for scalability. Features and homographies take an inordinate amount of time to compute; therefore, these processes are pre-computed offline. The online or real-time procedure then renders multiple images by stitching them together with the pre-computed homographies. The result is that an increase in the number of images or the amount of known geometry only increases the amount of offline calculations, not that of the online rendering.

The offline procedure finds SIFT features for each image and calculates a 3x3 homography between nearby images within the RANSAC framework [16], [30].

C. Online Real-Time Rendering

The online process loads the homographies from disk into memory to stitch relevant images to the "best" image chosen in Section V-A for a given view. The renderer performs a similar procedure to determine images considered to be neighbors by taking into account position and orientation relative to the "best" image. These neighbors are cached for future searches, limiting the search calculation to a single pass across all images.

The camera pose information for each image is available from the backpack localization results as described in Section IV. In addition to camera poses, the homography parameters such as the number of inliers and the residual error resulting from the homography compilation are taken into account in choosing neighboring images for stitching purposes. Inherently, these transformations assume that the images are coplanar, but in practice the scene is not always a flat environment. Therefore, homographies that transform the image by a rotation of more than 45 degrees of any axis are not applied as they likely correspond to non-planar scenes.

1) Culling and Intersection:

If we were to estimate a homography for each pair of images, the run time would become prohibitively long, i.e. on the order of $O(n^2)$ where n is the number of images. However, optimizations can be made by exploiting the known geometry of the environment. Specifically, the backpack localization results can be

used to generate a 3D point cloud as shown in Figure 8(a), and a resulting planar model as shown in Figure 8(b). An intersection test can then be used to detect occlusions due to planes, and to determine whether two images can be considered neighbors for the mosaicing step [40].

2) Alpha Blending:

Neighboring images can have inconsistent lighting or distortion that affect the mosaic; as such, the boundaries between images can be fairly pronounced, diminishing the visual appeal. We choose to use a variant of feathering to alpha blend images together in OpenGL. To do so, we divide each plane into a series of one pixel wide planes, and apply a triangular weighting function horizontally to the *i*th one pixel wide plane.

D. Image Based Rendering Results

We describe the capability of our proposed renderer on a T-shaped hallway with 2709 wall, ceiling, and floor images at 1338x987 pixels from 3 cameras. The rendering machine has 8 Intel Xeon 2.66 Ghz CPUs with 4GB of RAM running 64-bit Ubuntu 8.04 using an nVidia Quadro FX 4600 graphics card with 768MB of memory. The multithreaded renderer takes up to two hours to process SIFT features for 2709 images and up to six hours to find homographies among all such images. The renderer can display single camera images at approximately 20 frames per second. An initial cost of stitching images together reduces the frame rate to 10 frames per second for two stitched images. As expected, an inverse relationship exists between the number of images stitched together and the frame rate. The optimal field of view with high frame rate is around 5 images stitched together resulting in 5 frames per second.

The viewer can navigate the image based renderer using the keyboard and mouse to control translation and orientation respectively. Rotating the view allows the viewer to look up, down, and sideways, corresponding to each of the 3 cameras on the backpack: the top, bottom, and left cameras. In our current implementation, the displayed views are limited to the images taken along the path of the backpack because view interpolation has not yet been incorporated. With the estimated pose of the camera, each image is transformed to the world coordinate frame; the images are approximated to have straight lines and right-angle corners. A map of the environment from the plane fitted models directs the user to navigate throughout the scene. The goal of the navigation is to allow the user to quickly and efficiently view the walls, floor, and ceiling at any position within the scene.

A rendered video sequence for the T-shaped corridor can be found in [18].

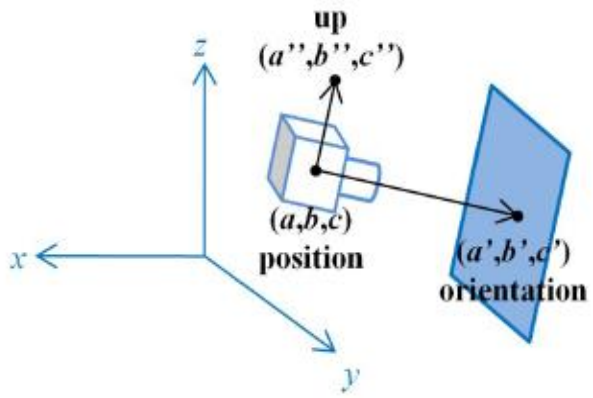


Fig. 13. Camera pose. The position vector is the camera's world coordinates. The orientation vector is the center of projection of the image. The up vector defines the rotation of the camera.

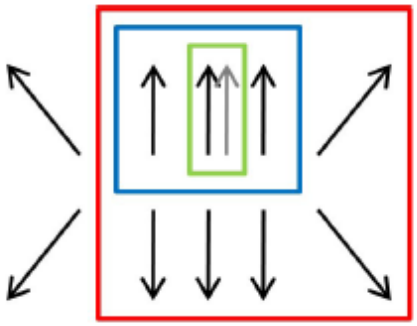


Fig. 14. Finding the best image to render. The grey vector is the viewer's pose and the black vectors represent camera poses. Red indicates neighboring images, blue similarly oriented images, and green closest image.

REFERENCES

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[2] D. Borrmann, J. Elseberg, K. Lingemann, A. Nuchter, and J. Hertzberg, "Globally consistent 3D mapping with scan matching," *Robotics and Autonomous Systems*, vol. 56, pp. 130–142, 2008.

[3] V. Pradeep, G. Medioni, and J. Weiland, "Visual loop closing using multi-resolution SIFT grids in metric-topological SLAM," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[4] M. Bosse and R. Zlot, "Continuous 3D scan-matching with a spinning 2D laser," in *Proc. of the IEEE international conference on Robotics and Automation*. Piscataway, NJ, USA: IEEE Press, 2009, pp. 4244–4251.

[5] F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2D range scans," *Journal of Intelligent and Robotic Systems*, vol. 18, pp. 249–275, 1994.

[6] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, 2007.

[7] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[8] M. Cummins and P. Newman, "Probabilistic appearance based navigation and loop closing," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2007.

[9] K. Granstrom, J. Callmer, F. Ramos, and J. Nieto, "Learning to detect loop closure from range data," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2009.

[10] G. Grisetti, S. Grzonka, C. Stachniss, P. Pfaff, and W. Burgard, "Efficient estimation of accurate maximum likelihood maps in 3D," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[11] G. Chen, J. Kua, S. Shum, N. Naikal, M. Carlberg, and A. Zakhor, "Indoor localization algorithms for a human-operated backpack," in *3D Data Processing, Visualization, and Transmission*, 2010.

[12] P. Newman and M. Cummins, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[13] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[14] C. I. Chow and C. N. Liu, "Approximating discrete probability distributions with dependence trees," *IEEE Transactions on Information Theory*, vol. 14, pp. 462–467, 1968.
[15] M. Carlberg, J. Andrews, P. Gao, and A. Zakhor, "Fast surface reconstruction and segmentation with ground-based and airborne LIDAR range data," in *Proceedings of the Fourth International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, 2008.
[16] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.
[17] A. Censi, "An accurate closed-form estimate of ICP's covariance," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2007.
[18] http://www-video.eecs.berkeley.edu/research/indoor.shtml.
[19] H. Shum and S. B. Kang, "A review of image-based rendering techniques," K. N. Ngan, T. Sikora, and M.-T. Sun, Eds., vol. 4067, no. 1. SPIE, 2000, pp. 2–13.
[20] L. McMillan and G. Bishop, "Plenoptic modeling: an image-based rendering system," in *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1995, pp. 39–46.
[21] D. G. Aliaga and I. Carlbom, "Plenoptic stitching: a scalable method for reconstructing 3D interactive walk throughs," in *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 2001, pp. 443–450.
[22] P. Debevec, Y. Yu, and G. Boshokov, "Efficient view-dependent image-based rendering with projective texture-mapping," Berkeley, CA, USA, Tech. Rep., 1998.
[23] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen, "Unstructured lumigraph rendering," in *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 2001, pp. 425–432.
[24] R. Koch, B. Heigl, and M. Pollefeys, "Image-based rendering from uncalibrated lightfields with scalable geometry," in *Proceedings of the 10th International Workshop on Theoretical Foundations of Computer Vision*. London, UK: Springer-Verlag, 2001, pp. 51–66.
[25] M. Uyttendaele, A. Criminisi, S. B. Kang, S. Winder, R. Szeliski, and R. Hartley, "Image-based interactive exploration of real-world environments," *IEEE Comput. Graph. Appl.*, vol. 24, no. 3, pp. 52–63, 2004.
[26] N. Snavely, S. M. Seitz, and R. Szeliski, "Modeling the world from internet photo collections," *Int. J. Comput. Vision*, vol. 80, no. 2, pp. 189–210, 2008.
[27] N. Snavely, R. Garg, S. M. Seitz, and R. Szeliski, "Finding paths through the world's photos," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 1–11, 2008.
[28] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3D," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 835–846, 2006.
[29] R. Szeliski, "Image alignment and stitching: a tutorial," *Found. Trends. Comput. Graph. Vis.*, vol. 2, no. 1, pp. 1–104, 2006.
[30] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.
[31] Florent Lafarge, Renaud Keriven, Mathieu Br´edif, Vu Hoang Hiep, "Hybrid multi-view Reconstruction by Jump-Diffusion", CVPR 2010.
[32] David Gallup, Jan-Michael Frahm, and Marc Pollefeys, "Piecewise Planar and Non-Planar Stereo for Urban Scene Reconstruction", CVPR 2010.
[33] Anne-Laure Chauve, Patrick Labatut, Jean-Philippe Pons, "Robust Piecewise-Planar 3D Reconstruction and Completion from Large-Scale Unstructured Point Data", CVPR 2010
[34] Norihiko Kawai, Tomokazu Sato and Naokazu Yokoya, ""Efficient Surface Completion Using Principal Curvature and Its Evaluation", IEEE Int. Conf. on Image Processing (ICIP2009), pp. 521-524, Nov. 2009.
[36] http://www.openscenegraph.org/projects/osg/wiki/Downloads
[37] http://www.bitmanagement.com/

[38] http://mgarland.org/software/qslim.html
[39] http://www-video.eecs.berkeley.edu/research/indoor/
[40] T. Liu, M. Carlberg, G. Chen, Jacky Chen, J. Kua, A. Zakhor, "Indoor Localization and Visualization Using a Human-Operated Backpack System," to be presented at the 2010, International Conference on Indoor Positioning and Indoor Navigation, Zurich, Switzerland, September, 2010.